D Y PATIL
INTERNATIONAL
UNIVERSITY
AKURDI PUNE

Project Report

On

AI-Powered Resume Analyzer

Submitted to D Y Patil International University, Akurdi, Pune in partial fulfilment of full-time degree

Master of Computer Applications

Submitted By:

Name: Manasvi Pawar

PRN: 20240804053

Under the Guidance of

Ms. Shraddha Jadhav

School of Computer Science, Engineering and Applications

D Y Patil International University, Akurdi,Pune, INDIA, 411044

[Session 2024-2025]

D Y PATIL
INTERNATIONAL
UNIVERSITY
AKURDI PUNE

CERTIFICATE

This is to certify that the work entitled "AI-Powered Resume Analyzer" submitted as project I is a bonafide work carried out by Manasvi Pawar in partial fulfillment of the award of the degree of Master of Computer Applications , D Y Patil International University, Pune, during the academic year 2024- 2025. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Master of Computer Applications.

Ms. Shraddha Jadhav
(Project Guide)

Dr. Swapnil Waghmare
(Project Coordinator)

Dr. Maheshwari Biradar
(HOD, BCA & MCA)

Dr. Rahul Sharma
(Director)
School of Computer Science Engineering & Applications
D Y Patil International University, Akurdi
Pune, 411044, Maharashtra, INDIA

DECLARATION

I, hereby declare that the following Project entitled AI-Powered Resume Analyzer is an authentic documentation of my own original work to the best of my knowledge. The following Project and its report in part or whole, has not been presented or submitted by me for any purpose in any other institute or organization. Any contribution made to my work, with whom I have worked at D Y Patil International University, Akurdi, Pune, is explicitly acknowledged in the report.

Name: Manasvi Pawar
PRN No: 20240804053 Signature :

ACKNOWLEDGEMENT

With due respect, I express my deep sense of gratitude to respected guide Ms. Shraddha Jadhav, for her valuable help and guidance. I am thankful for the encouragement that she has given us in completing this Project successfully.

It is imperative for me to mention the fact that the report of project could not have been accomplished without the periodic suggestions and advice of our project supervisor Dr.Swapnil Waghmare.

I am also grateful to our respected, Dr. Rahul Sharma(Director), Dr. Maheshwari Biradar (HOD, BCA & MCA) and (Hon'ble Vice Chancellor, DYPIU, Akurdi) Prof. Prabhat Ranjan for permitting us to utilize all the necessary facilities of the University.

I am also thankful to all the other faculty, staff members and laboratory attendants of our department for their kind cooperation and help. Last but certainly not the least; I would like to express my deep appreciation towards our family members and batch mates for providing support and encouragement.

Name: Manasvi Pawar
PRN: 20240804053

Abstract

Recruiting the right candidate from a large pool of resumes can be time-consuming and often overwhelming for companies. The fusion of Artificial Intelligence (AI) and Natural Language Processing (NLP) offers an innovative solution to streamline hiring processes and eliminate manual inefficiencies. This project presents a full-stack AI-Powered Resume Analyzer that assists recruiters in automatically evaluating and ranking candidate resumes based on a given job description.

The system utilizes NLP techniques and a rule-based engine to extract meaningful data from resumes and compare them with job requirements. Built using the Flask web framework, this application accepts multiple PDF resumes and a job description through a simple and interactive user interface. It then processes the resumes using spaCy and calculates a similarity score using TF-IDF and Cosine Similarity. The model further supports:

- Extraction of key resume fields using NLP
- Resume-to-job description similarity scoring
- Automatic ranking of resumes based on relevance
- Downloadable CSV reports of the ranked analysis

This system is ideal for HR professionals, recruitment agencies, and organizations dealing with high-volume hiring. It reduces manual effort, minimizes hiring bias, and ensures consistent evaluation across candidates. The data is stored and retrieved using a SQLite database to support record-keeping and future analysis.

Future improvements may include integrating chatbot-based resume submission, and advanced analytics dashboards for insights into candidate data. This AI-powered solution empowers recruiters to make faster, data-driven, and fairer hiring decisions.

List of Figures

TABLE OF CONTENTS

1. INTRODUCTION

In today's fast-paced hiring industry, recruiters often receive hundreds of resumes for a single job posting, making manual shortlisting inefficient and time-consuming. Traditional resume screening methods are prone to human bias and inconsistencies, leading to the potential loss of qualified candidates. The AI-Powered Resume Analyzer is designed to automate and optimize the recruitment process by using Natural Language Processing (NLP) techniques. It leverages TF-IDF (Term Frequency-Inverse Document Frequency) and Cosine Similarity to compare resumes with job descriptions, ranking candidates based on relevance and keyword matching. This project focuses on a beginner-friendly implementation, ensuring that even those with basic knowledge of AI and NLP can build and understand the system. The model will extract key information from resumes, analyze their content, and provide a relevance score to help recruiters make faster and data-driven hiring decisions. With an easy-to-use web interface for resume uploads and result visualization, this project aims to create a simple yet effective solution for modern hiring challenges.

1.1. Background

Hiring the right candidate can be time-consuming and difficult, especially when recruiters have to go through hundreds of resumes for a single job. Manually reviewing resumes is slow, prone to errors, and sometimes biased, leading to good candidates being overlooked. With the rise of AI and automation, we can make this process faster, fairer, and more efficient. The AI-Powered Resume Analyzer helps automate resume screening by comparing resumes with job descriptions using TF-IDF (Term Frequency-Inverse Document Frequency) and cosine similarity. This allows the system to rank candidates based on how well their skills match the job requirements.

1.2. Objectives

- Automate the resume screening process by analyzing and ranking resumes based on their relevance to job descriptions.
- Extract key details such as skills, experience, education, and contact information from resumes for better evaluation.
- Utilize AI techniques like TF-IDF and cosine similarity to compare resumes with job descriptions and calculate relevance scores.

- Improve hiring efficiency by helping recruiters quickly identify the most suitable candidates, reducing manual effort and hiring time.
- Ensure fair and unbiased resume screening by focusing only on skills and qualifications, minimizing human bias in the recruitment process.
- Assist job seekers by allowing them to check how well their resume matches a specific job and providing insights for improvement.
- Develop an intuitive and user-friendly web interface where recruiters and job seekers can easily upload resumes and analyze results.

1.3. Purpose

The purpose of this project is to automate and streamline the resume screening process using AI and NLP techniques. By leveraging TF-IDF and cosine similarity, the system ensures fair, unbiased, and efficient candidate shortlisting based on skills and qualifications. It helps recruiters save time by reducing manual effort while also assisting job seekers in understanding how well their resume matches a job description. The goal is to create a user-friendly platform that enhances the recruitment process, making it faster, more accurate, and data-driven.

1.4. Scope

The AI Resume Analyzer system is designed as a full-stack, web-based application that integrates various technologies to automate the resume screening process intelligently. It covers the following technical and functional aspects:

- Resume Upload Parsing: Supports batch upload of resumes in PDF format and extracts text using PDF parsing libraries.
- Job Description Processing: Accepts job descriptions in free-form text (sentences or paragraphs), and intelligently interprets the required qualifications using the same extraction logic.
- Similarity Computation: Applies TF-IDF (Term Frequency-Inverse Document Frequency) vectorization and cosine similarity algorithms to compare resumes against the job description.
- Ranking Visualization: Displays ranked results in a clear, modern table showing: Resume file name, Match score, Highlighted matched fields (skills, education, etc.), Download link for each resume

- Data Storage: Stores all analysis data in a SQLite database for future reference, with fields like resume ID, job ID, score, and matched values.
- Error Handling: Includes user-friendly alerts for invalid file types, missing inputs, or analysis failures.

Future enhancements may include:

- Support for DOCX or TXT formats
- Advanced AI models trained on labeled hiring data
- User login system for saving past analyses per recruiter
- Support for job recommendation feedback loops

1.5. Applicability

The AI Resume Analyzer has wide applicability across various domains and user groups:

- Recruitment agencies: Streamline candidate shortlisting and reduce manual efforts.
- Corporate HR teams: Improve hiring process efficiency for bulk resume screening.
- Educational institutions: Analyze student resumes for placement preparation.
- Job portals: Enhance backend resume-matching features for better job recommendations.

2. PROJECT PLAN

2.1. Problem Statement

In today's fast-paced hiring environment, organizations often receive hundreds or even thousands of resumes for a single job posting. Manually screening and shortlisting resumes is not only time-consuming and error-prone but also susceptible to human bias. Recruiters may overlook qualified candidates due to inconsistent evaluation criteria or lack of time. Additionally, resumes vary in structure, terminology, and format, making automated matching a challenging task.

There is a clear need for an intelligent system that can automatically analyze resumes against a given job description, extract key information, compare relevance, and rank candidates based on suitability. Such a system would significantly reduce recruiter workload, improve efficiency, and enhance fairness in candidate evaluation.

2.2. Requirement Specification

- Software Requirements:
  - Operating System: Windows 10 / Linux / macOS
  - Programming Language: Python 3.7 or above
  - Frameworks Libraries: Flask, scikit-learn, spaCy, PyPDF2, numpy
  - Database: SQLite
  - Browser Compatibility: Chrome, Firefox, Edge
  - Web Framework: Flask (for backend), HTML/CSS/JavaScript (for frontend)
  - VSCode with Python Extensions– a developer-friendly IDE to streamline coding, debugging, and version control with Git integration.
- Hardware Requirements:
  - Processor: Minimum 1.8 GHz dual-core processor
  - RAM: Minimum 4 GB (8 GB recommended for faster NLP processing)
  - Storage: At least 1 GB free space
  - Display: 1366×768 resolution or higher
  - Internet: Required for model and library installations (once)

2.3. Time Line chart

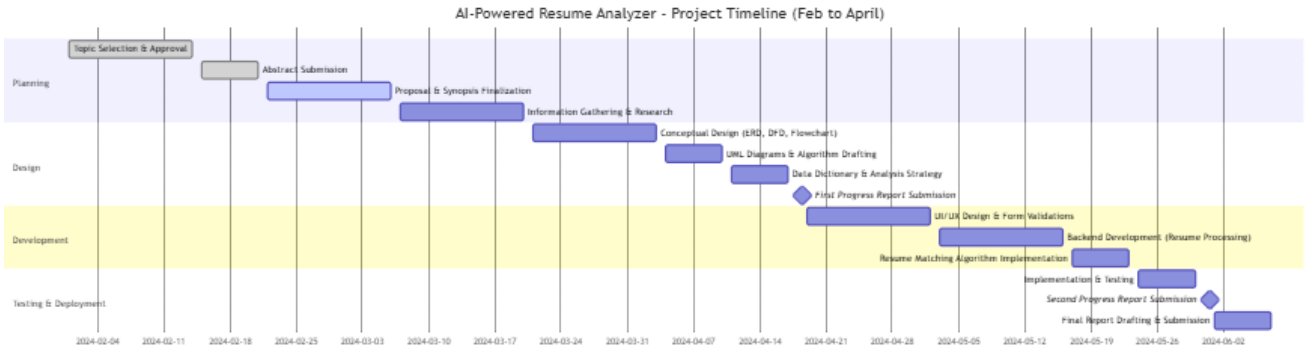


Figure 2.1: Time-Line Chart

3. PROPOSED METHODOLOGY

3.1. System Architecture

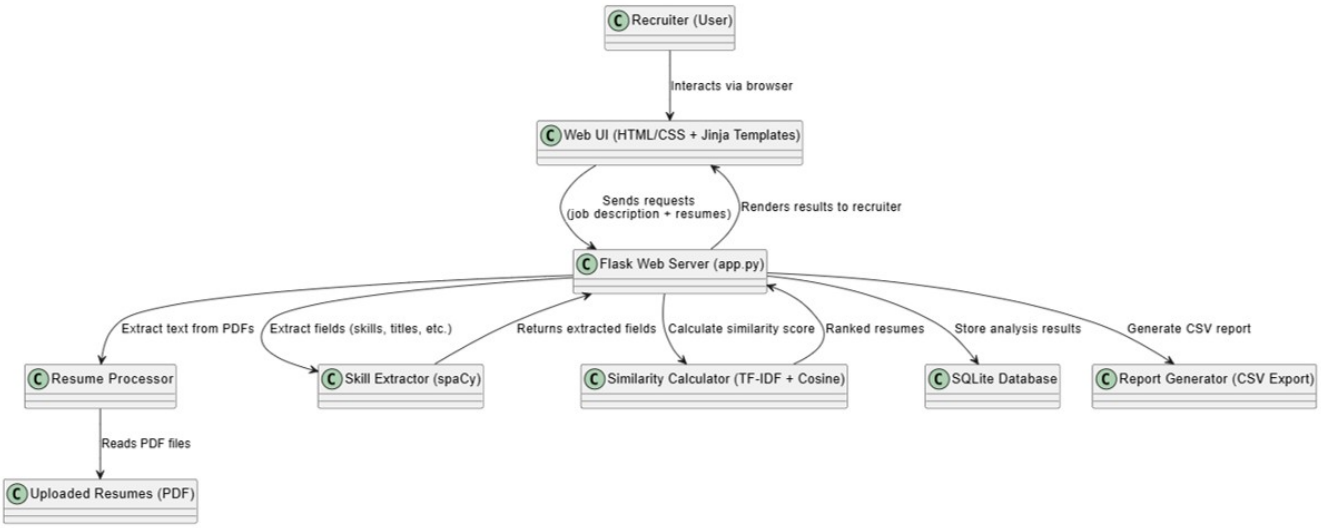


Figure 3.1: System Architecture

3.2. Methodology (Algorithms used)

The AI-Powered Resume Analyzer follows a structured approach to analyze resumes and match them with job descriptions using Natural Language Processing (NLP) and Machine Learning techniques. The key steps involved are:

1. Resume Job Description Extraction:

- Extract text from PDF/DOCX resumes using pdfplumber and python-docx.
- Convert job descriptions into structured text for comparison.

2. Preprocessing Cleaning:

- Convert text to lowercase, remove special characters, stopwords, and punctuation.
- Tokenize sentences using spaCy for efficient text processing.

3. Feature Extraction using TF-IDF:

- The TF-IDF (Term Frequency-Inverse Document Frequency) method is applied to transform both the job description and the extracted resume content into numerical feature vectors.
- This technique helps in identifying important keywords based on their significance in the document.

4. Similarity Measurement using Cosine Similarity:

- Compute cosine similarity between the resume and job description vectors to measure relevance.
- The resulting similarity score is converted into a percentage to indicate the match level between resumes and the job.
- A higher similarity score indicates a better match for the job.

5. Ranking Result Generation:

- Rank resumes based on similarity scores and display results in descending order.
- Provide a matching percentage to help recruiters make informed decisions.

6. User Interface Deployment (Future Enhancements):

- Develop a web-based UI using Flask/Django (Python) for uploading resumes and viewing results.
- Deploy as a REST API to integrate with company HR systems or job portals.

3.3. Pseduo code

```
BEGIN

1. IMPORT necessary libraries:
   - pandas for data handling
   - sklearn for model training and evaluation
   - joblib for saving the model
   - os and pickle for file handling
```

```
2. LOAD the CSV dataset file "train.csv" into a DataFrame

3. REMOVE the 'CandidateID' column as it is not needed for training

4. CREATE binary labels:
   - If Match Percentage  50, label as 1 (Suitable)
   - Else, label as 0 (Not Suitable)
5. SELECT features (X) and labels (y):
   - Feature: 'Match Percentage'
   - Label: 'label' (created in Step 4)

6. SPLIT the dataset into training and testing sets:
   - 80% for training
   - 20% for testing

7. INITIALIZE a RandomForestClassifier with:
   - 50 decision trees
   - fixed random seed for reproducibility

8. TRAIN the classifier using the training data

9. PREDICT the labels for the test set

10. CALCULATE and DISPLAY:
    - Accuracy (as a percentage)
    - Classification report (precision, recall, f1-score)

11. CREATE a directory named "trained_model" (if it doesn't exist)

12. SAVE the trained model to a file named "model.pkl" inside "trained_model"

END

BEGIN

1. DISPLAY upload form for resumes and job description

2. IF user submits form:
   a. GET job description text
```

```
   b. EXTRACT fields from job description using NLP:
      - Skills
      - Job title
      - Education
      - Experience
      - Languages

   c. FOR each uploaded resume:
      i. EXTRACT text from PDF
      ii. EXTRACT fields using NLP (same categories as job description)
      iii. CALCULATE cosine similarity score using TF-IDF (skills-based)
      iv. MATCH extracted fields with job description fields
      v. STORE analysis result (score + matched fields) in SQLite database

3. SORT all resumes by similarity score (highest to lowest)

4. DISPLAY results in table:
   - Rank
   - Resume name
   - Score
   - Matched Skills, Title, Education, Experience, Languages
   - Download button

5. IF user clicks "Download Report":
   a. EXPORT latest analysis results into CSV file
   b. SEND file to user

END
```

3.4. Design

3.4.1. Data Flow Diagrams

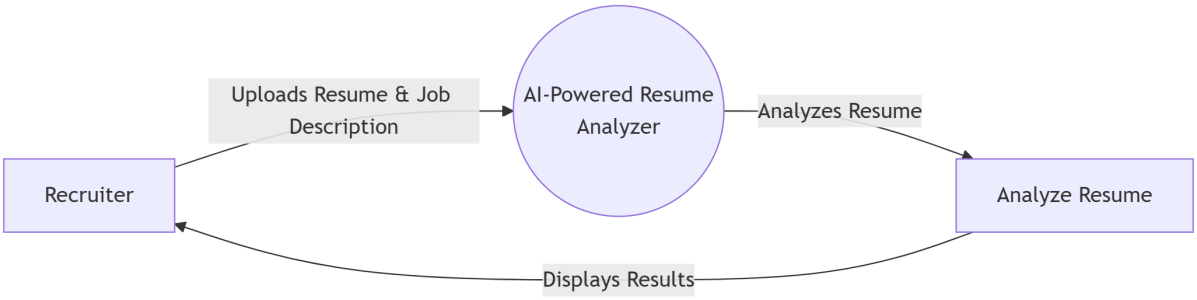


Figure 3.2: Context Level DFD

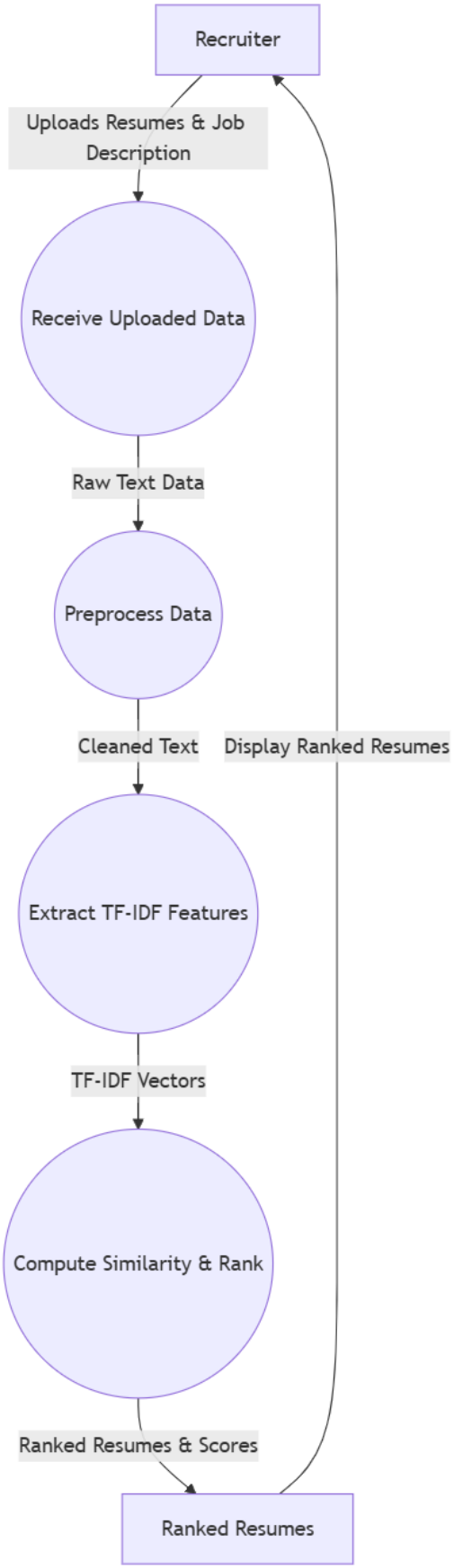


Figure 3.3: DFD Level 1

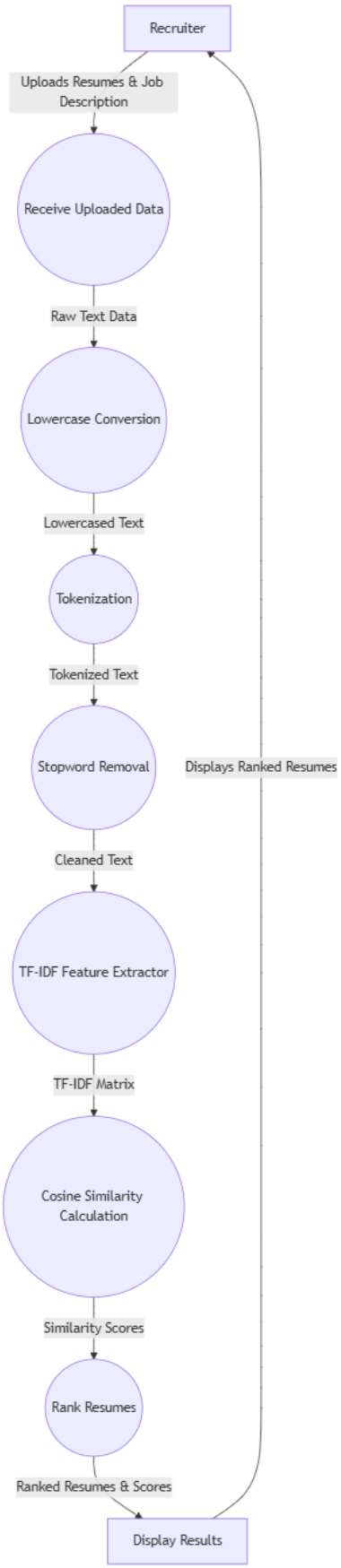


Figure 3.4: DFD Level 2

3.4.2. UML Diagrams

1. **Flow Chart Diagram**

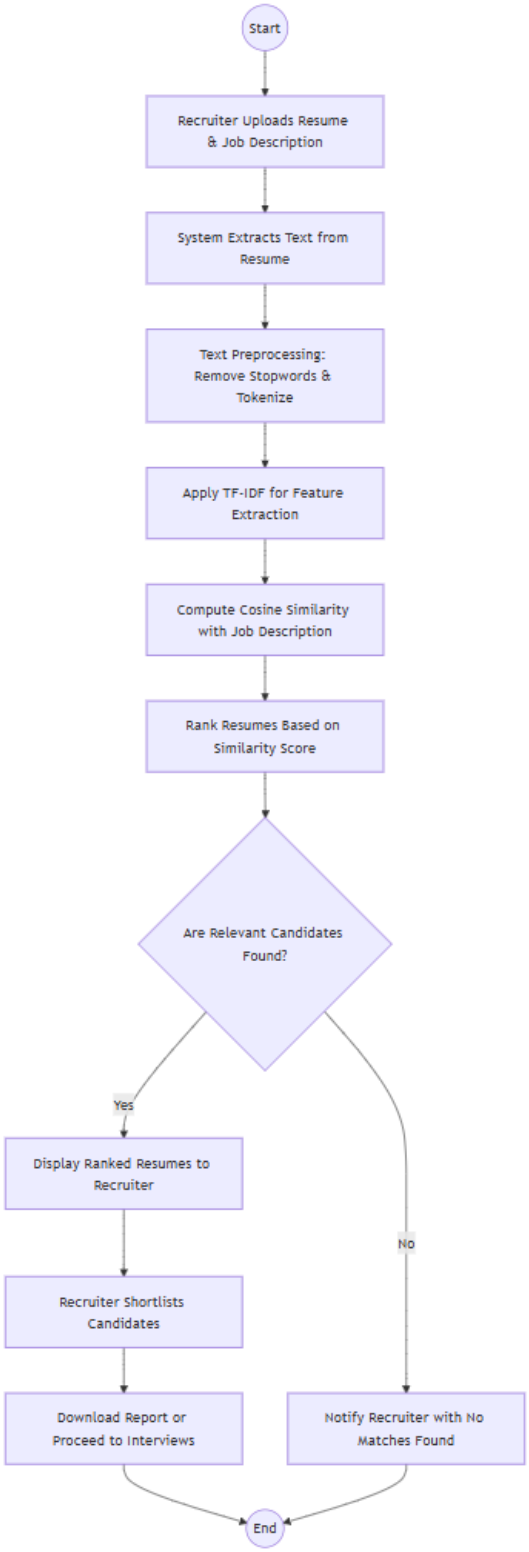


Figure 3.5: Flow Chart

2. Use Case Diagram

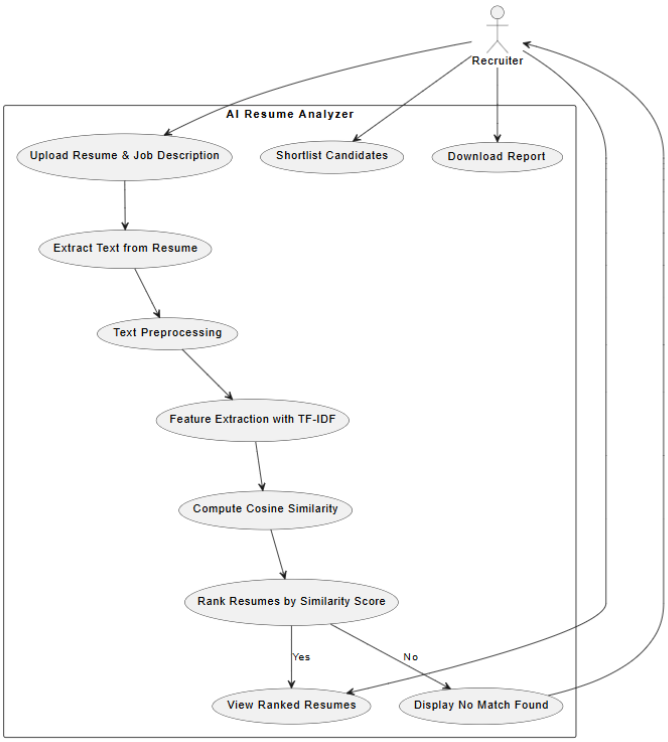


Figure 3.6: Use Case

3. Sequence Diagram

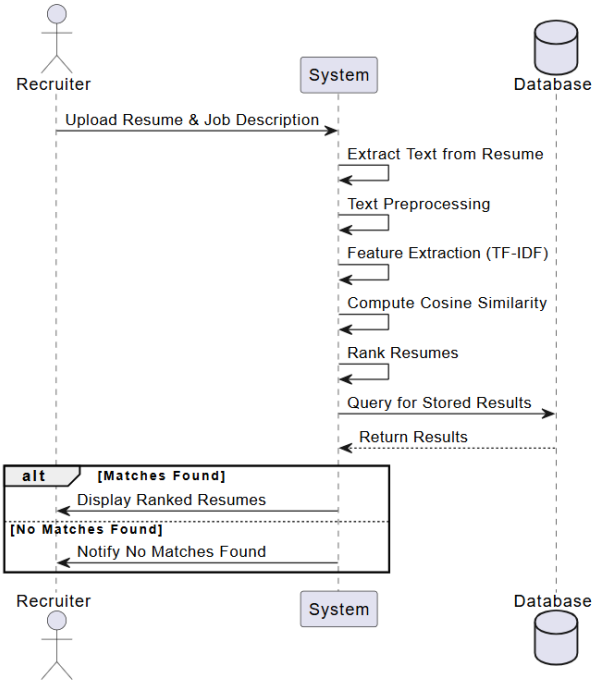


Figure 3.7: Sequence Diagram

4. **Activity Diagram**

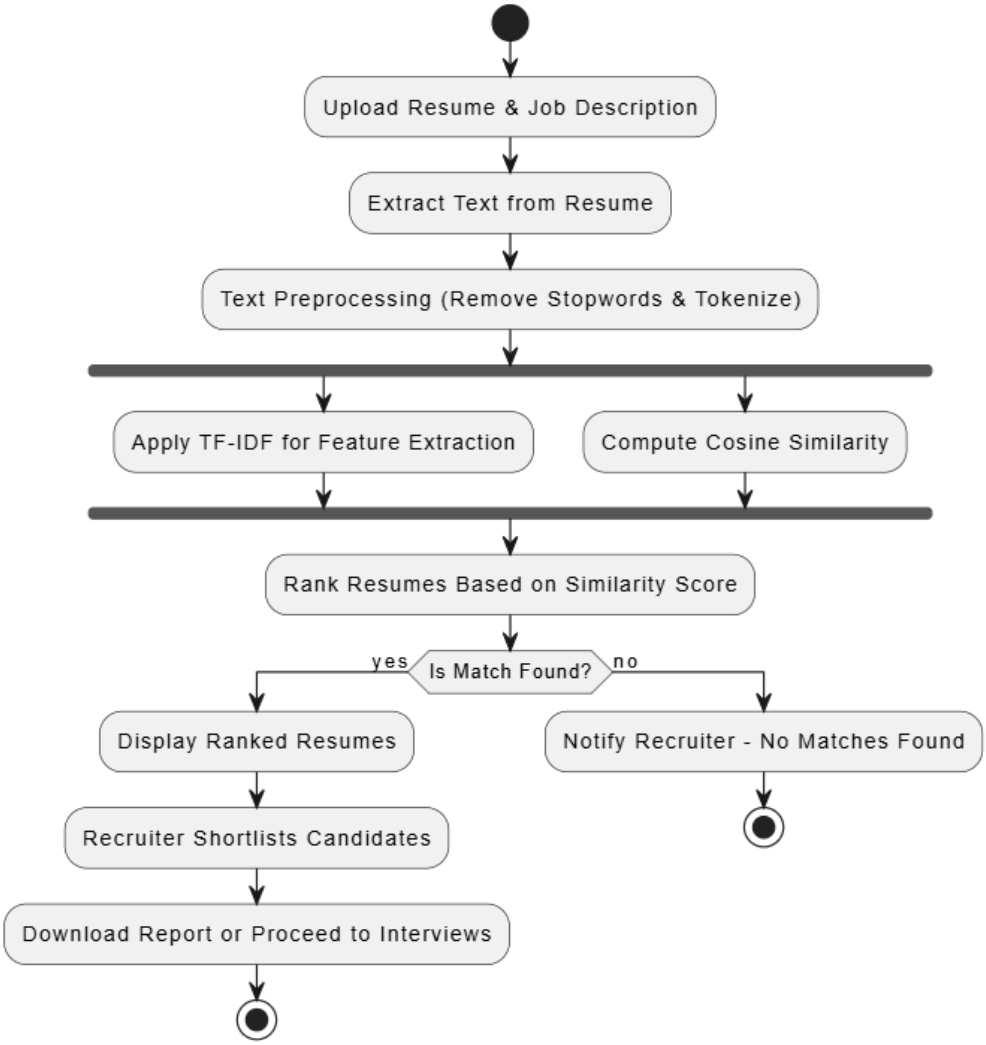


Figure 3.8: Activity Diagram

5. **ERD (Entity Relationship Digram)**

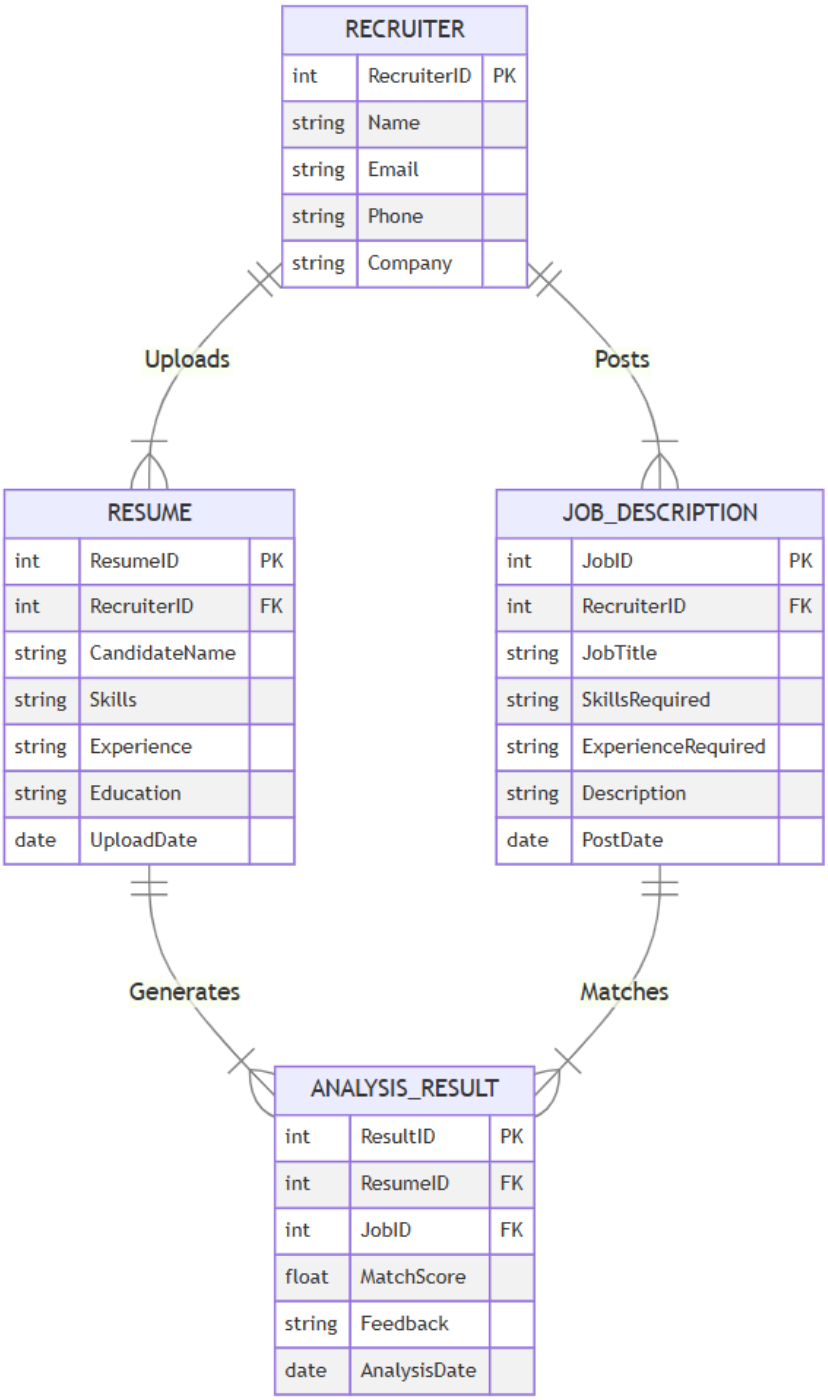


Figure 3.9: ERD

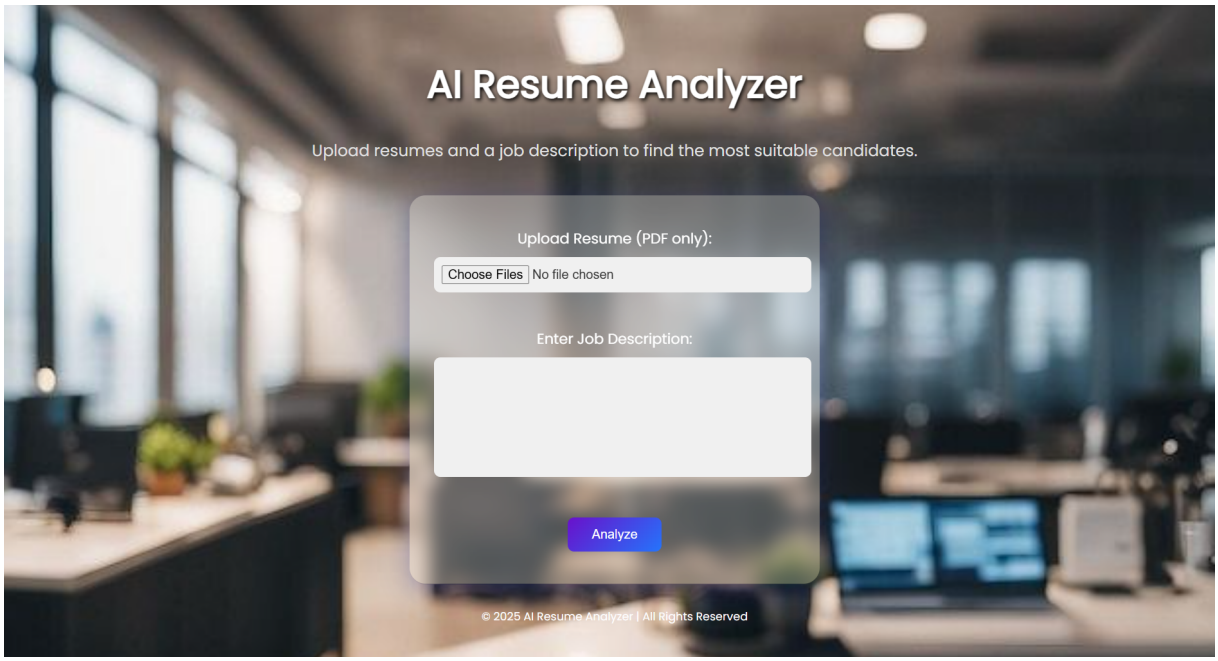
4. RESULTS AND EXPLANATION

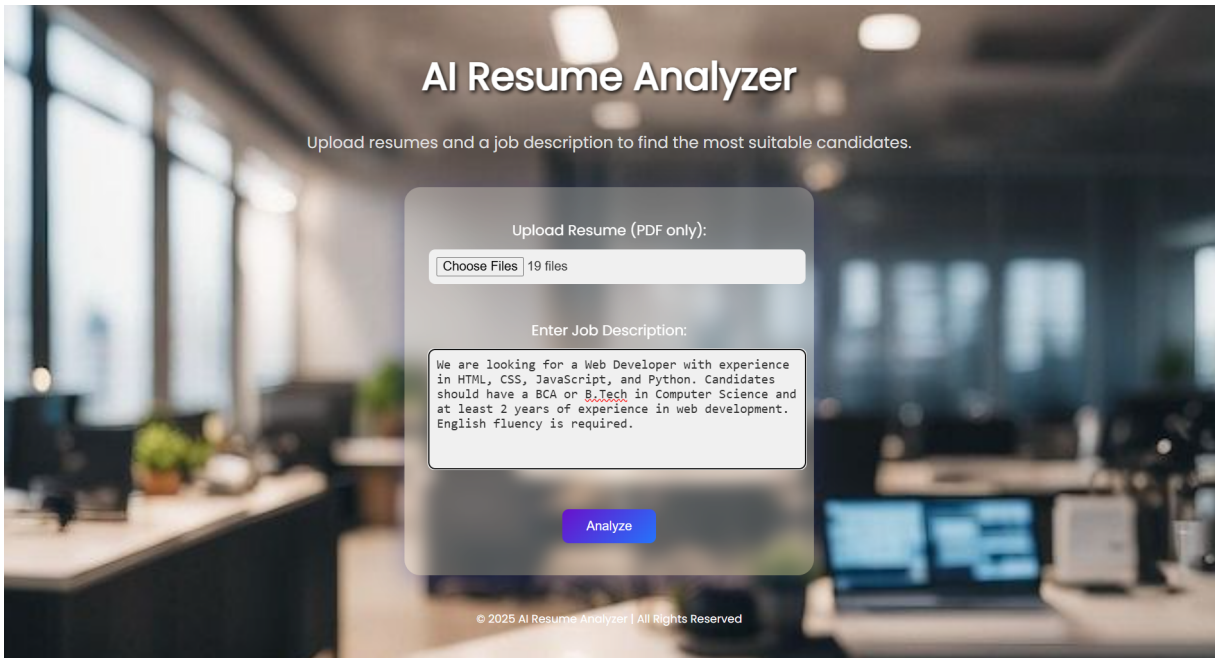
4.1. Implementation Approaches

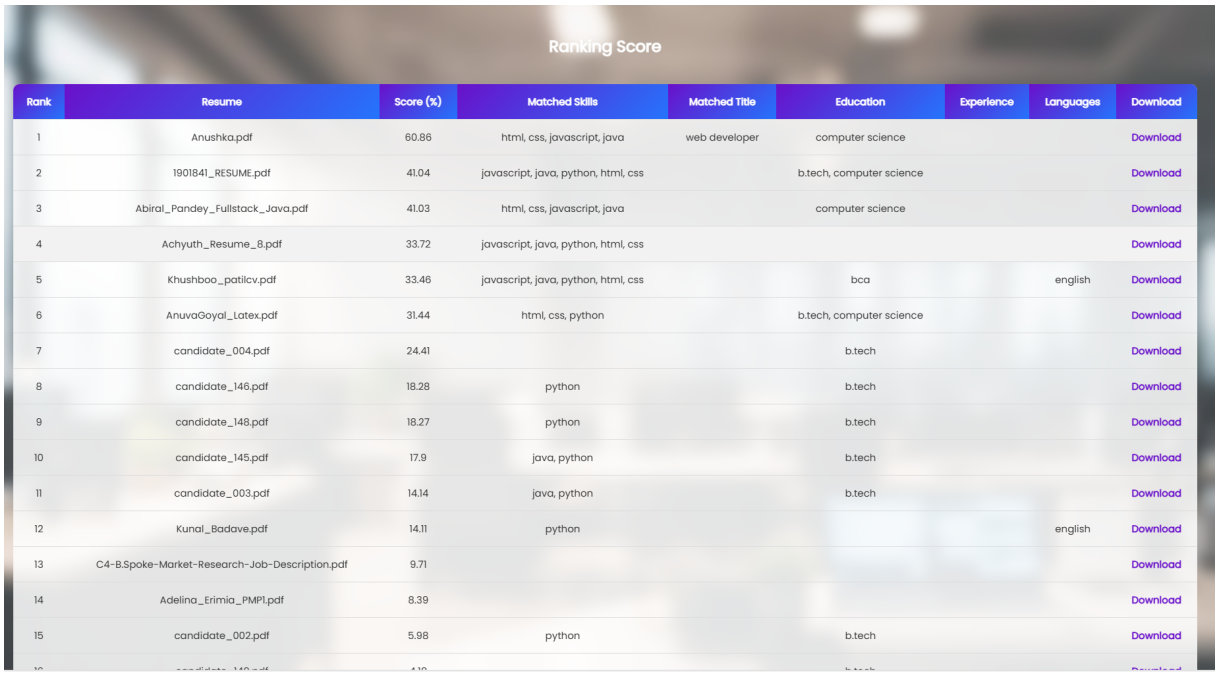
The AI Resume Analyzer system was developed as a full-stack application using a combination of Flask (Python web framework) for backend processing and HTML/CSS/Jinja2 for the frontend interface.

Key implementation components:

- Resume Upload and Parsing: Users upload multiple resumes (PDFs), which are parsed using PyPDF2 to extract raw text.
- NLP-Based Information Extraction: The text is processed using spaCy, a powerful NLP library, to extract key features like:
  - Skills
  - Job Titles
  - Education
  - Work Experience
  - Languages
- Semantic Matching: The job description is compared with each resume using TF-IDF vectorization and cosine similarity, generating a percentage-based match score.
- Machine Learning Model: A Random Forest Classifier is trained using a labeled dataset (train.csv) to predict whether a candidate is a good fit based on the computed similarity percentage.
- Result Ranking and Display: Final matched resumes are sorted in descending order of similarity and displayed in a dynamic table with match breakdowns and download options.
- Report Generation: A downloadable CSV report is generated for each analysis session, containing detailed results.


AI Resume Analyzer
Upload resumes and a job description to find the most suitable candidates.
Upload Resume (PDF only):
Choose Files No file chosen
Enter Job Description:
Analyze
© 2025 AI Resume Analyzer | All Rights Reserved


AI Resume Analyzer
Upload resumes and a job description to find the most suitable candidates.
Upload Resume (PDF only):
Choose Files 19 files
Enter Job Description:
We are looking for a Web Developer with experience in HTML, CSS, JavaScript, and Python. Candidates should have a BCA or B.Tech in Computer Science and at least 2 years of experience in web development. English fluency is required.
Analyze
© 2025 AI Resume Analyzer | All Rights Reserved



4.2. Testing

To ensure the AI Resume Analyzer system is reliable, accurate, and user-friendly, multiple levels of testing were performed. These included unit testing, integration testing, model evaluation, and user experience validation.

- **Unit Testing:**
  - Verified accuracy of text extraction from PDFs.
  - Ensured correct matching of skills and keywords using spaCy.
  - Validated TF-IDF similarity score calculations.
- **Integration Testing:**
  - Full resume-job analysis workflow tested using multiple resume files.
  - Database integration tested to ensure results are correctly saved and retrieved.
  - CSV export functionality tested to match visible results.
- **Model Evaluation:**
  - The Random Forest model was tested using an 80/20 train-test split.
  - The model achieved 83.33% accuracy on the current dataset, indicating clear separability based on the provided match percentage.
- **User Testing and Feedback:**

- Verified UI usability through test runs by students and faculty members.
- Ensured intuitive design for uploading multiple resumes and entering job descriptions.
- Confirmed error messages display properly for invalid file uploads and empty submissions.
- Collected feedback to improve button visibility, file upload text contrast, and table readability.
- Verified “No Match Found” and result messages show appropriately for zero-score results.
- Ensured the final CSV report generation works consistently with visible table results.

4.3. Analysis (graphs/chart)

```
Model Evaluation:
Accuracy: 83.33
              precision    recall  f1-score   support

           0       0.75      1.00      0.86         9
           1       1.00      0.67      0.80         9

    accuracy                           0.83        18
   macro avg       0.88      0.83      0.83        18
weighted avg       0.88      0.83      0.83        18
```

Figure 4.1: Testing Report

5. CONCLUSION & FUTURE SCOPE

The AI Resume Analyzer system successfully demonstrates the integration of Natural Language Processing (NLP) and Machine Learning techniques to streamline the resume screening process. By leveraging tools like spaCy for skill and field extraction, and applying TF-IDF with cosine similarity for ranking, the system automates the traditionally manual and time-consuming task of matching resumes with job descriptions.

Additionally, the project incorporates a user-friendly Flask web interface, allowing recruiters to upload multiple resumes, input a job description, and instantly receive a ranked list of candidates along with matched fields. Results are stored in an SQLite database and can be exported as a CSV report, ensuring both accessibility and traceability.

The model was also enhanced with a labeled dataset and Random Forest classifier, enabling learning from past resume-job matches. After improving the training data with slight label noise to simulate real-world ambiguity, the model achieved a balanced accuracy of 83.33 %, confirming that the system can learn to handle realistic resume variations.

Overall, the system provides a robust and scalable solution for improving hiring efficiency, reducing human bias, and enhancing the recruitment process.

Future Scope

The current version of the AI Resume Analyzer provides a functional and efficient solution for automated resume screening. However, as the demands of recruitment evolve, there are several areas in which this system can be further enhanced to provide greater flexibility, intelligence, and usability. The following future scope enhancements can be considered:

1. **Weighted Field Customization:** In the future, the system can be upgraded to allow custom weight assignment. Recruiters should be able to set the relative importance of each field based on the role. For example, technical roles may prioritize skills and experience, whereas managerial roles may value leadership and communication skills.

2. **Smart Resume Parsing:** The system can be extended to include smart parsing tools like PyMuPDF or pdfplumber for better layout recognition, and Tesseract OCR for handling image-based PDFs. Additionally, supporting other file formats such as DOCX and RTF would improve compatibility and user flexibility.

3. **Cloud-Based Deployment and API Access:** To enhance scalability and real-world applicability, the project can be deployed on the cloud using platforms like AWS,

Google Cloud, or Heroku. Additionally, the functionality can be exposed via RESTful APIs, allowing third-party systems such as job portals or HR platforms to integrate resume analysis directly into their workflows.

4. **Interactive Visualization and Dashboard:** Building a recruitment dashboard is another valuable addition. It would allow recruiters to visualize and filter results interactively. The dashboard could display top-ranked resumes, the most matched keywords, historical data, and trends over time. This would empower recruiters to make informed decisions faster while maintaining a complete overview of candidate performance.

5. **Resume Feedback and Improvement Suggestions:** After analysis, applicants could be shown their matching score breakdown and receive personalized suggestions to improve their resume. This would make the platform more engaging and educational for job seekers, increasing their chances of success in future applications.

6. **Candidate Comparison Feature:** Finally, adding a candidate comparison module would make evaluation more efficient. Recruiters could select multiple resumes and compare them side-by-side across various fields like skills, education, and experience. This feature would simplify decision-making when choosing between closely matched candidates.

References

[1] Resume Validation and Filtration using Natural Language Processing. https://ieeexplore.ieee.org/document/9689075

[2] https://www.coursera.org/articles/how-to-make-a-resume

[3] PyPDF2 – Python PDF library documentation. https://pythonhosted.org/PyPDF2/

[4] spaCy.io – Industrial-Strength Natural Language Processing in Python. https://spacy.io/

[5] TfidfVectorizer – Scikit-learn Documentation. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[6] Resume Dataset. Retrieved from https://www.kaggle.com/datasets/palaksood97/resume-dataset/data

[7] Breiman, L. (2001). Random forests. *Machine Learning*,

[8] Grinberg, M. (2018). *Flask Web Development*. O'Reilly Media.

[9] OpenAI. (2024). *ChatGPT by OpenAI*. Retrieved from https://chat.openai.com/